vertex.py

```python
from utils import *
from matrix import Matrix, M, Ro_Any_Matrix

class Vertex:
    def __init__(self, _x : float, _y : float, _z : float) -> None:
        self.x : float = _x
        self.y : float = _y
        self.z : float = _z
        self.V : tuple[float, float, float] = self.set_tuple()

    def set_tuple(self) -> tuple[float, float, float]:
        return (self.x, self.y, self.z)

    def length(self) -> float:
        return (self.x ** 2 + self.y ** 2 + self.z ** 2) ** (1 / 2)

    def distance(self, _v) -> float:
        return ((self.x - _v.x) ** 2 + (self.y - _v.y) ** 2 + (self.z - _v.z) ** 2) ** (1 / 2)

    def draw(self) -> None:
        glBegin(GL_POINTS)
        glVertex3f(self.x, self.y, self.z)
        glEnd()

    def translate(self, _vector : list[float]) -> list[float]:
        new_vertex : Matrix = M['T'](_vector) * Matrix([*self.V, 1], _is_vector=True)
        self.x = new_vertex.V[0]
        self.y = new_vertex.V[1]
        self.z = new_vertex.V[2]
        self.V = new_vertex.V

    def rotate(self, _position, _angle : float, _axis : str) -> None:
        new_vertex : Matrix = M[_axis](_angle) * Matrix((self - _position).V, _is_vector=True)
        self.x = new_vertex.V[0] + _position.x
        self.y = new_vertex.V[1] + _position.y
        self.z = new_vertex.V[2] + _position.z
        self.V = new_vertex.V

    def rotate_any(self, _position, _yaw : float, _pitch : float, _roll : float) -> None:
        new_vertex : Matrix = Ro_Any_Matrix(_yaw, _pitch, _roll) * Matrix((self - _position).V, _is_vector=True)
        self.x = new_vertex.V[0] + _position.x
        self.y = new_vertex.V[1] + _position.y
        self.z = new_vertex.V[2] + _position.z
        self.V = new_vertex.V

    def __sub__(self, _v):
        return Vertex(self.x - _v.x, self.y - _v.y, self.z - _v.z)

    def __mul__(self, _mul):
        if isinstance(_mul, (int, float)):
            return Vertex(self.x * _mul, self.y * _mul, self.z * _mul)

    def __repr__(self) -> str:
        return f'({self.x}, {self.y}, {self.z})'
```